Typing-Teaching System For Beginning Programmers

CSU Bakersfield
School of Natural Sciences, Mathematics, and Engineering

Andres Zamora, D'Angelo Felix, Juan Orozco, Joshua Chavez

Intro:

This web application is designed to support motivated and aspiring computer programmers who lack basic computer and typing literacy skills. The lack of these skill sets challenge aspiring computer programmers because these are essential and foundational skill sets for the computer science field. Our web application bridges that computer literacy void through the processes of an interactive, engaging, and beginner level student-paced platform for motivated individuals who yearn to learn the foundational computer and typing skill sets.

Objective:

Our objective is to develop a web application that can improve a user's typing skill competency, and teach the user the basics of programming languages, such as C++. The overall goal is to make it user-friendly, engaging, and interactive for the user to enjoy challenging themselves in building their own typing proficiencies and learning the foundations of coding.

Computer Science Background:

The idea of developing a student-paced typing program with the hybrid modality of an engine and language was initially unfamiliar to our C++ backgrounds. However, Unity's coding language and it's engine were well documented for us to move forward with a clear direction and to determine a measurable scope of our project.

Unity Implementation:

Unity's interface was used to create the menu, games, and quizzes. Buttons, text elements, and sprites are all created using Unity's interface. However, they can be controlled to perform specific tasks using scripts. Scripting was an essential part of making everything work together properly.

Scripting in C#:

We used the C# language for our scripts. Initially, adapting to this language was difficult because we were not familiar with C# and formatting the scripts became a challenge. We tried making one big script per game to handle functionality, but it did not work the way we wanted. So, we overcame this challenge by designing multiple scripts that worked together to make a game work. Then, there was the task of connecting each script together. After some adjustments, the scripts collaborated on making the game run.

Vector Implementation:

There was an issue where we could not get the scene screen to match the game screen in Unity. It would perform well on the scene screen. However, the game screen is what the user sees. Since they didn't match, the ship would travel to random points on and off screen. Moreover, each word runs on its own vector. We need each word to fall at random positions at a specified field with a fixed-increasing speed and a delay between each word. These components work well with the game screen to an extent. After adjusting between different render modes, the two screens matched, and the ship finally took the position of the words.

Format:

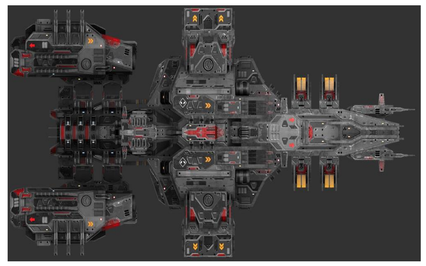
There are lessons that help the user learn the basics of the programming language C++. Each lesson contains a set of slides and a quiz. Each set focuses on a specific subject of the language and comes equipped with a quiz to challenge the user on what they learned from the slides. After they pass the quizzes, they are able to play mini-games that include the syntax of what they learned from the lessons. Each game will challenge their speed and knowledge of the language. If the user has trouble with basic keyboard skills outside the programming world, then there are typing exercises that help the user learn their way around the keyboard.

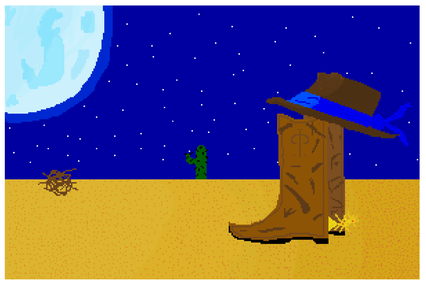
Influences:

For developing a student-paced typing web application program, we researched various flashcard websites. The two most influential websites were flashcardmachine.com and studyblue.com/ because we learned from their programming format for their lessons and quizzes user-interface. We needed to find a balance between the amount of information we researched, and the amount of examples in question design for our user audience. The questions needed to be clear, concise, and informative for a beginner-level and younger audience. For our theme, we played many typing games, and one of our favorites was Z-Type typing.com/student/game/ztypeThis game had us decide on to go with the space theme.

Art:

For the art we decided to use 2D pixel art. For the backgrounds, we had to learn how to create this art. We studied and practiced in creating color palettes and implementing shading to make it a little more detailed.





Conclusion:

Our goal was to create a game with two main objectives, teach the user how to type properly and also to teach the user the basics of coding. The age group we were targeting elementary ages of 7-10 years old. However, anyone is able to play our game and learn about programming in C++. We designed an interactive game that is both entertaining and educational.